UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/827,197	04/06/2001	Richard D. Webb	1930.0070003	5381

26111 7590 03/25/2004

STERNE, KESSLER, GOLDSTEIN & FOX PLLC
1100 NEW YORK AVENUE, N.W.
WASHINGTON, DC 20005

EXAMINER	
YIGDALL, MICHAEL J	

ART UNIT	PAPER NUMBER
2122	2

DATE MAILED: 03/25/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/827,197	WEBB, RICHARD D.	
	Examiner	Art Unit	
	Michael J. Yigdall	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *06 April 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-27* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-27* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *06 April 2001* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-27 are pending and have been examined. The priority date considered for the application is 4 August 2000.

Drawings

2. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description: 115, 125, 135, 137, 215, 222, 223, 224, 225 and 226. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

3. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they include the following reference sign(s) not mentioned in the description: 200, 230, 314, 316, 318, 320, 340 and 410. A proposed drawing correction, corrected drawings, or amendment to the specification to add the reference sign(s) in the description, are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Claim Objections

4. Claim 16 is objected to because of the following informalities: The phrase "namespace-scoped global" should perhaps be replaced with --namespace-scoped global variables--. Appropriate correction is required.

Claim Rejections - 35 USC § 101

5. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

6. Claims 1-16 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claims 1-16 are recited as a programming language, which is simply a representation of the rules used to define a computer program, and is considered an abstract idea or an arrangement of data *per se*. Descriptive material that cannot exhibit any functional interrelationship with the way in which computing processes are performed does not constitute a statutory process, machine, manufacture or composition of matter. See MPEP § 2106(IV)(B)(1)(b).

Claim Rejections - 35 USC § 112

7. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

8. Claim 6 is rejected under 35 U.S.C. 112, first paragraph, because the specification, while being enabling to describe the programming language with the syntax BNF supported by the Scheme language (see paragraph 0021 on page 6), does not reasonably provide enablement for the specific syntax recited in the claim. The specification does not enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the invention commensurate in scope with these claims. The limitations recited in claim 6 comprise formal syntax rules for the programming language that are not disclosed in the specification.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

10. Claims 1- 4, 7-9 and 15 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 6,480,856 to McDonald et al. (hereinafter McDonald).

With respect to claim 1, McDonald discloses an extensible, object-oriented, portable programming language that permits centrally defined resource management, wherein an object expressed by the language can be simple or compound (see the title and abstract), and

wherein a simple object comprises the following attributes: an object name (see column 10, lines 4-7), an object type (see column 10, lines 4-7, which shows an object "shape" or type), a version (see column 13, lines 23-36, which shows an index representing a version of an object), defined accessibility (see column 19, lines 52-55, which shows accessibility defined by the implementation), and

wherein a compound object comprises the following attributes: an object name (see column 10, lines 4-7), a base object (see column 23, lines 46-52, which shows a parent or base object), a field (see column 10, lines 57-63, which shows properties or fields), defined accessibility (see column 19, lines 52-55, which shows accessibility defined by the

implementation), and a persistence property (see column 23, lines 15-24, which shows sharing memory structures among objects, i.e. persistence).

With respect to claim 2, McDonald further discloses the limitation wherein said compound object further comprises attributes selected from the group consisting of: a version, a child object, a parameter, a namespace, a C++ abstract base type, a volatile property, an external property, an inferior property, and a ccdoc operator (see FIG. 7 and column 10, lines 57-63, which shows attributes in the form of properties or parameters).

With respect to claim 3, McDonald further discloses the limitation wherein said simple object can be emulated as an enumeration object (see column 11, lines 5-15, which shows indexing the members of an object in a table, in the same manner as an enumeration object).

With respect to claim 4, McDonald further discloses the limitation wherein said field comprises state information (see column 10, lines 54-56, which shows state information such as visibility and bounds).

With respect to claim 7, McDonald further discloses the limitation wherein said language can express a nested list that comprises the following atomic elements (see column 18, lines 39-50, which shows a list of properties nested within a programming construct):

keywords (see column 18, lines 39-50, which shows keywords such as "HAS");

names of auto-generated elements (see column 18, lines 39-50, which shows names of generated properties); and

literals (see FIG. 14, which shows that literals are used to specify the property values).

With respect to claim 8, McDonald further discloses the limitation wherein said literals of the nested list are selected from the group consisting of Booleans, numbers, and strings (see column 18, lines 39-50, which shows the "bool" or Boolean type, the "int" or number type, and the "text" or string type).

With respect to claim 9, McDonald further discloses the limitation wherein said auto-generated elements are selected from the group consisting of objects and field names (see column 18, lines 39-50, which shows that the elements comprise property or field names).

With respect to claim 15, McDonald further discloses the limitation wherein the objects are selected from the group consisting of C++ classes, namespaces, templates, and constant values (see column 6, lines 58-60, which shows that objects are associated with containers, i.e. namespaces).

11. Claims 17 and 19-25 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 6,427,228 to Wigger.

With respect to claim 17, Wigger discloses a system for describing structure of programming languages (see the abstract, which shows a system for specifying or describing objects of a programming language), comprising:

(a) a high-level programming language (see column 6, lines 51-57, which shows using high-level programming languages such as Java and C++);

(b) an extensible, object-oriented programming language for describing said high-level programming language (see column 8, lines 20-30, which shows using Extensible Markup Language to specify a meta file associated with the high-level language); and

(c) a programming tool for converting said object-oriented programming language (see column 4, lines 22-39, which shows combining or converting the meta file to produce an object-oriented source code output file).

With respect to claim 19, Wigger further discloses the limitation wherein input and verification parameters are specified in said extensible and object-oriented descriptive programming language (see column 9, lines 7-18, which shows specifying parameters in the meta file that serve as input values).

With respect to claim 20, Wigger further discloses the limitation wherein said programming tool is a compiler (see column 6, lines 41-47, which shows implementing the system in a compiler).

With respect to claim 21, Wigger further discloses the limitation wherein said programming tool is a translator (see column 6, lines 41-47, which shows implementing the system in a compiler; note that a compiler is a translator).

With respect to claim 22, Wigger discloses a method for describing computer programs by retaining meta-information about program elements (see the abstract, which shows a system for specifying or describing objects of a programming language in a meta file), thereby allowing optimization and functionality on multiple hardware and software platforms (see column 6, lines

51-57, which shows using the Java language, and column 8, lines 20-30, which shows using XML; note that the Java language and XML are platform-independent and have functionality on multiple hardware and software platforms), comprising the following steps:

(a) creating a first program using a high-level programming language (see column 6, lines 51-57, which shows using high-level programming languages such as Java and C++; see also FIG. 4A, which shows an exemplary program);

(b) creating a second corresponding program using an extensible, object-oriented programming language to describe the high-level source code (see column 8, lines 20-30, which shows using Extensible Markup Language to specify the meta file associated with the high-level language; see also FIG. 3, which shows an exemplary program); and

(c) converting the second corresponding program into a form of the high-level programming language (see column 4, lines 22-39, which shows combining or converting the meta file to produce an output file in the form of a high-level programming language; see also FIG. 5A, which shows an exemplary output file).

With respect to claim 23, Wigger further discloses the limitation wherein the form is machine-executable (see column 6, lines 41-47, which shows that the program is compiled and executed, and is therefore machine-executable).

With respect to claim 24, Wigger further discloses the limitation wherein the form is high-level programming language (see column 4, lines 22-39, which shows producing an output file in the form of a high-level programming language).

With respect to claim 25, Wigger further discloses the limitations wherein results of said step (a) and said step (b) are placed into one file (see column 4, lines 22-39, which shows combining the source program and the meta file to produce an output file), and further comprising the steps of:

(d) copying said second corresponding program from the file (see column 14, lines 47-51, which shows copying code from the meta file); and

(e) combining said second corresponding program with the form of the high-level source code (see column 4, lines 22-39, which shows combining the source program and the meta file to produce an output file).

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

13. Claims 5 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over McDonald, as applied to claims 4 and 15 above, respectively, in view of what is well known in the art of object-oriented programming.

With respect to claim 5, McDonald further discloses the limitation wherein said state information comprises the following attributes: a field name (see FIG. 7, which shows property or field names), a field type (see column 18, lines 39-50, which shows field types), an initial

default value (see column 27, lies 7-13, which shows a default value), accessibility (see column 19, lines 52-55, which shows accessibility), an override property (see column 27, lines 7-13, which shows the functional equivalent to an override property), an automatic set function (see column 16, lines 34-50, which shows set functions), an automatic get function (see column 16, lines 34-50, which shows get functions).

Although McDonald discloses an object-oriented programming model (see the title and abstract), McDonald does not expressly disclose the limitation wherein said state information comprises a construct property, a destruct property, and a ccdoc operator.

However, examiner takes Official Notice that constructors, destructors and comments are well known in the art. Constructors and destructors are well-known methods for creating new objects in memory and destroying objects already existing in memory, respectively. Comments are well known for providing source code documentation within a program. Note that a comment used for documentation is analogous to a ccdoc operator.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include, in the state information of McDonald, a construct attribute, a destruct attribute and a ccdoc operator, because such features are well known in the art of object-oriented programming for managing objects in memory and providing documentation.

With respect to claim 16, although McDonald discloses an object-oriented programming model (see the title and abstract), McDonald does not expressly disclose the limitation wherein said constant values are selected from the group consisting of enums, class static variables, and namespace-scoped global.

However, examiner takes Official Notice that enumerations, class static variables and namespace-scoped global variables are well known in the art for the purpose of defining constants. For example, an enumeration or enum may be used to specify a sequence of named values, and variables of different scopes may be used to specify values depending on where in the program such values are to be accessed.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use enums, class static variables or namespace-scoped global variables in the system of McDonald, as appropriate, in order to define constant values.

14. Claims 6 and 10-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over McDonald, as applied to claim 1 above, in view of *Chez Scheme User's Guide* by R. Kent Dybvig (hereinafter Dybvig).

With respect to claim 6, although McDonald discloses an object-oriented programming model (see the title and abstract), McDonald does not expressly disclose a syntax described by the following syntax BNF:

letter ::= "A" | "B" | ... | "Z",

number ::= "1" | "2" | ... | "9",

atomic_symbol ::= letter atom_part,

atom_part ::= empty | letter atom_part | number atom_part,

empty ::= "",

S_expression ::= atomic_symbol | "("S_expression"."S_expression")" | list,

list ::= "("S_expression # S_expression")".

However, Dybvig discloses the formal syntax of the Scheme programming language with Chez Scheme extensions using Backus-Naur Form (see "Formal Syntax," pages 2-4, which shows the BNF of expressions, identifiers and data as recited in the claim).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to describe the programming language of McDonald using the BNF taught by Dybvig, for the purpose of formally defining its syntax.

With respect to claim 10, although McDonald discloses the limitation wherein said language can express hierarchically structured packages (see column 10, lines 4-19, which shows that objects are hierarchically structured, in that one can be converted to another based on the subset or superset relationship between the two sets of properties), McDonald does not expressly disclose the limitation wherein the packages are selected from the group consisting of applications and libraries.

However, Dybvig discloses modules or packages that may be selected and imported from other applications or built-in libraries, for the purpose of organizing programs written by many developers (see "Syntactic Extension," pages 13-20, sections 9.3 and 9.4).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to select packages from applications and libraries in the system of McDonald, as taught by Dybvig, for the purpose of organizing programs written by many developers.

With respect to claim 11, the combination of McDonald and Dybvig further discloses the limitation wherein said packages can be defined by: (library (name, nameoflibrary)...) (see

Dybvig, "Syntactic Extension," page 13, which shows the syntax for defining a package, and page 14, which shows the notation for multiple levels of parentheses).

With respect to claim 12, the combination of McDonald and Dybvig further discloses the limitation wherein said packages can be defined by: (library, nameoflibrary...) (see Dybvig, "Syntactic Extension," page 13, which shows the syntax for defining a package).

With respect to claim 13, the combination of McDonald and Dybvig further discloses the limitation wherein said packages can be defined by: (application (name, nameofapplication)...) (see Dybvig, "Syntactic Extension," page 13, which shows the syntax for defining a package, and page 14, which shows the notation for multiple levels of parentheses).

With respect to claim 14, the combination of McDonald and Dybvig further discloses the limitation wherein said packages can be defined by: (application, nameofapplication...) (see Dybvig, "Syntactic Extension," page 13, which shows the syntax for defining a package).

15. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Wigger, as applied to claim 17 above, in view of U.S. Pat. No. 6,675,370 to Sundaresan.

With respect to claim 18, Wigger further discloses the limitation wherein compiler pragmas are automatically added to the system (see column 6, lines 41-47, which shows a preprocessing step prior to compilation; see also block 450 in FIG. 4A, which shows compiler directives or pragmas as indicated with the "%" character).

Although Wigger discloses statements used for documentation (see FIG. 4A, which shows comments as indicated by the "//" characters), Wigger does not expressly disclose the limitation wherein copyright text and CCDoc directives are automatically added to the system.

However, Sundaresan discloses automatically generating documentation by adding Javadoc directives (see column 3, lines 1-10), which are analogous to CCDoc directives, using an extensible language (see column 3, lines 36-43). Note that Sundaresan further discloses a preprocessor for producing browsable documentation for high-level languages that do not already have that capability, such as C++ (see column 5, lines 43-49).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to add CCDoc directives to the system of Wigger, as taught by Sundaresan, for the purpose of automatically providing source code documentation.

Furthermore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use the comments in the output file of Wigger (see FIG. 5A) for copyright text, in order to identify the owner of the source code.

16. Claim 26 is rejected under 35 U.S.C. 103(a) as being unpatentable over Wigger, as applied to claim 25 above, in view of U.S. Pat. No. 6,546,549 to Li.

With respect to claim 26, Wigger does not expressly disclose the limitation wherein the file is a header file.

However, Li discloses producing header files (see files F4, F5 and F6 in FIGS. 1 and 2, and column 6, lines 8-24, which shows include files or header files), in a source code transformation system for generating stub methods adaptable to a plurality of execution

environments (see column 4, lines 18-22). Note that Li further discloses using a high-level programming language as well as extensible object-oriented programming language (see column 5, lines 27-57, which shows using C++ and extended C++).

It would have been obvious to one of ordinary skill in the art at the time the invention was made for the output file in the system of Wigger to be used as a header file, as taught by Li, for the purpose of adapting programs to a plurality of execution environments.

17. Claim 27 is rejected under 35 U.S.C. 103(a) as being unpatentable over Wigger in view of Li as applied to claim 26 above, and further in view of *Microsoft Press Computer Dictionary, Third Edition* (hereinafter Dictionary).

With respect to claim 27, although Li discloses including header files in C++ programs (see column 6, lines 8-24), the combination of Wigger and Li does not expressly disclose the limitation wherein the header file comprises the following sections: definitions, user preamble, user pre-object, user member, user post-object and user post-amble.

However, header files are well known in the art to comprise definitions of data types and declarations of variables used by a program (see Dictionary, page 229, "header file"). For example, a header file may define a class (i.e., a definitions section) and a number of functions or methods (i.e., a user member section), and have comments to describe the header file and specify how class objects and functions are to be used (i.e., user preamble, pre-object, post-object and post-amble sections).

It would have been obvious to one of ordinary skill in the art at the time the invention was made for the header file in the system of Wigger and Li to comprise definitions, user

preamble, user pre-object, user member, user post-object and user post-amble sections, because it is well known in the art that header files may comprise such sections to describe information used by a program.

Conclusion

18. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (703) 305-0352. The examiner can normally be reached on Monday through Friday from 8:00am to 4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2122

mjy
March 16, 2004

TUAN DAM
SUPERVISORY PATENT EXAMINER